Full Stack Nordeus Challenge

QA

Maša Cucić

1. Bug Descriptions and Fixes

Here are some of the bugs I encountered during the development of this project.

- **Bug 1: Incorrect Tile Click Detection**

  Description: I encountered an issue with click detection on the map, specifically when clicking on individual tiles. The problem became evident when clicking on tile (0,0), as the click was occasionally misinterpreted as being outside the map, while other times it was correctly detected. I found that the click detection was inconsistent and depended on the exact position of the click within the tile. The root cause was a misalignment between the tile positions and the click detection logic. For instance, the map was generated starting at (0,0), with each 1x1 tile centered on this point. As a result, the coordinates of the first tile ranged from (-0.5, 0.5). This caused clicks on the tile to sometimes be detected as outside the map (due to negative coordinates), and at other times, as valid clicks on tile (0,0) if the click occurred within the positive bounds of the tile.

  How I fixed it: To address this, I applied an offset to the world position of the click, specifically adding half of the tile size (tileSize/2). This adjustment guarantees accurate detection of a tile's coordinates, ensuring they align perfectly with the player's click.

- **Bug 2: Island Partitioning**

  Description: The issue occurred because I was initiating a DFS search for every unvisited tile. However, this approach caused the same island to be detected as multiple islands because the search would spread out from different starting points. When the search encountered a previously visited tile, it mistakenly treated it as a separate island, even though it was part of the same island.

  How I fixed it: To fix this, I changed the search to only start from unvisited tiles that represent land. This way, the search only begins when a new island is encountered,

ensuring that the DFS spreads only from the new island and avoids incorrectly detecting multiple islands.

- **Bug 3: Clicks Still Registered After Game Over**

  Description: The issue occurred because clicks were still being registered even after the game was over, allowing the player to continue interacting with the game. Specifically, clicking on the map after the game ended, such as when clicking the "Play Again" button, would trigger actions like reducing health and causing unintended behavior.

  How I fixed it: To resolve this, I used game states. I ensured that clicks are only registered when the game state is set to "Playing." This way, after the game ends, the game state is either "Win" or "Lose," and clicks are no longer registered since it's no longer in a "Playing" state.

- **Bug 4: Overly Random Map Generation**

  Description: The problem arose because I initially generated the map heights randomly, leading to an overly chaotic map. The random heights resulted in a map with a large number of small, disconnected islands, and unrealistic height transitions.

  How I fixed it: To resolve this, I researched and implemented Perlin noise to generate more realistic map heights. By adjusting parameters like scale (which controls the level of detail in the generated terrain), as well as xOffset and yOffset (which allow for different random map generations each time), I was able to create smoother, more natural-looking terrain with consistent elevation transitions. This made the map look much more realistic and visually appealing.

2. Testing Methods

Some of the testing methods I would apply in the projects of other contestants are:

- **Input testing**

  Testing various types of inputs, including valid, invalid, edge-case values and extreme input values, to ensure the game responds correctly to all possible user entries.

- **UI testing**
  Verifying the responsiveness and accuracy of user interactions, such as clicks on the map, on the edges of the map, on buttons, and in input fields, ensuring that all UI elements function as expected.
- **Functionality testing**
  Verifying that the map is generated correctly based on different user inputs and ensuring that the island with the highest average height is accurately identified across a variety of maps.
- **Special cases testing**
  Testing specific edge cases, such as clicks near the borders of the map or islands, to confirm that the game behaves as intended in these scenarios.

3. Improvements & Features for the Ideal Version

To start, I would focus on general design enhancements, such as adding animations to make the game more visually engaging. Additionally, I would introduce different levels of gameplay, each varying in aspects like map dimensions, tile size, height range, and other parameters.

Then, I would try to make the game as interesting as possible. If I could implement any extension of this game I imagine, I would create a multiplayer game where players compete to be the first to click on a specific island, earning points based on the number of islands and fields they control. The winner is the player with the most points at the end.

At the start of the game, each player chooses a color. The first island they click on becomes "conquered" and is outlined in their chosen color. Then, the game continues with the selection of the next island, but only among those that have not yet been conquered.

To make the game more interesting, each new task for conquering an island could be based on different criteria displayed on the screen, such as: "find the island with the most snow," "find the island with the most greenery," "find the lowest island," "find the largest island," "find the smallest island," and so on.

To make the game even more dynamic, I would introduce an additional feature: the player who conquers an island can swap two tiles on the map. This can result in the rearrangement of islands — large islands might be split into smaller ones, or some islands could be merged.

Additionally, if a player fails to win *n* consecutive rounds (i.e., they are not the first to click on the desired island *n* times in a row), one of their previously conquered islands becomes

"neutral" and is no longer controlled by any player. This island re-enters the game and becomes available for conquest again.

I would also introduce different difficulty levels, based on map size, as well as the number and complexity of criteria for selecting an island. Each level would feature a mix of easier and more challenging criteria. For example, an easier task might be "find the island with the most greenery," while a harder one could be "find the island that is the farthest from all others."

In the early game, players are rewarded for conquering as many separate islands as possible. This encourages quick expansion. However, as the game progresses, the focus shifts toward merging conquered islands into a single, larger territory. Players are encouraged to use the tile-swapping feature to merge their islands, as creating the largest island becomes the main goal by the end of the game. The winner is the player with the highest total score in the end.

4. Factors Impacting the Solution

Here are some factors that could potentially have a negative or unpredictable impact on my solution:

- **Tile size**

Increasing the tile size creates gaps between the tiles. While click detection should continue to work properly, this change could cause the map to overlap with other UI elements, as its position is determined by a single starting point. Larger tiles would make the map significantly larger, potentially extending beyond the screen. To solve this, I would add a frame around the map and scale it to fit within the frame, ensuring that the map remains fully visible and contained within the screen, regardless of the tile size, and preventing it from overlapping with other elements.

- **Map size**

The overall size of the map, in terms of the number of tiles, could influence how the tiles are arranged. Similar to the issue with tile size, a larger map might affect the layout, causing the tiles to shift in ways that complicate the click detection or interaction.

- **User Interaction**

Rapid clicking on the map by players can lead to unintended detection errors, such as selecting incorrect tiles or missing clicks entirely. Also, rapid changes of map, and inserts of random values could cause unpredictable behaviour as well.

- **Number of lives**

If the player had a different number of lives, the hearts displayed on the screen would need to be adjusted accordingly, so my current implementation wouldn't function as expected. To address this, I could place, for example, 10 invisible hearts on the screen, and based on the number of lives (up to 10), the corresponding number of hearts would be made visible.

- **Changes in height range**

If the maximum height is set to a value greater than 1000, my game would color all tiles with heights above 1000 in white, which is likely not the desired outcome. I could resolve this by scaling the color ranges to match the actual height range. Additionally, the color legend would no longer align with these changes. To fix this, I would need to update the text next to the color legend image accordingly.